Estimating the Deformability of Elastic Materials using Optical Flow and Position-based Dynamics

Püren Güler, Karl Pauwels, Alessandro Pieropan, Hedvig Kjellström and Danica Kragic

***Abstract*— Knowledge of the physical properties of objects is essential in a wide range of robotic manipulation scenarios. A robot may not always be aware of such properties prior to interaction. If an object is incorrectly assumed to be rigid, it may exhibit unpredictable behavior when grasped. In this paper, we use vision based observation of the behavior of an object a robot is interacting with and use it as the basis for estimation of its elastic deformability. This is estimated in a local region around the interaction point using a physics simulator. We use optical flow to estimate the parameters of a position-based dynamics simulation using meshless shape matching (MSM). MSM has been widely used in computer graphics due to its computational efficiency, which is also important for closed-loop control in robotics. In a controlled experiment we demonstrate that our method can qualitatively estimate the physical properties of objects with different degrees of deformability.**

I. INTRODUCTION

The ability to interact with and grasp objects is an integral part of an autonomous robot system. To handle an object appropriately and predict its behavior during interaction, it is of great importance to know its physical properties. Interactions with non-rigid objects are especially not easy to predict unless their deformation properties are known. Without this information, a robot may break or permanently deform an object, *e.g.* when grasping a milk container, a robot can break the container or spill the content. However, due to the large variability among objects, object properties cannot always be assumed to be known beforehand. It is therefore crucial to have a mechanism that enables the robot to learn such properties by observing the behavior of the object during the interaction.

To this end, we present a method that employs visual observation together with a physical simulation to estimate the deformability of objects. This is in the spirit of [1], where it is shown that in order to properly manipulate deformable objects, a robot needs to observe its environment, and predict the objects' deformation by modeling their behavior through simulation.

In our previous work [2], we showed that the content of a container can be inferred by observing the deformation of the object using multi-sensor fusion of visual and tactile data. However the visual tracking component of the system relied on a model of the object that assumes rigidity [3] and the divergence of the observed data from this rigid model was used to define deformability. The rigid model used by the tracker limited the amount of deformation the tracking system could handle. Here we use a deformable model that provides a better match with the observed scene and that enables us to model the temporal change in the shape more effectively.

The authors are with the Computer Vision and Active Perception Lab, Center for Autonomous Systems, School of Computer Science and Communication, KTH Royal Institute of Technology, Stockholm, Sweden, {puren,kpauwels,pieropan,hedvig,dani}@kth.se.
This work was supported by the Swedish Foundation for Strategic Research and the EU project RoboHow.Cog (FP7-ICT-288533). The GPUs used for this research were donated by the NVIDIA Corporation.

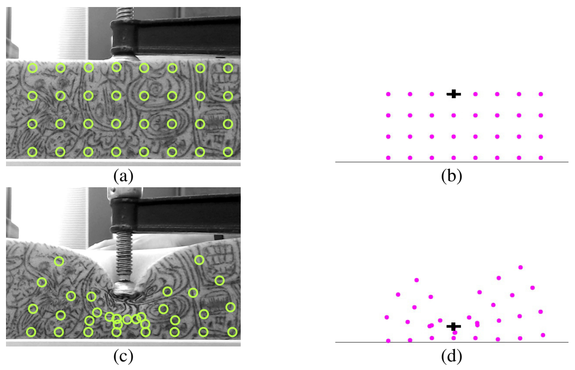


Fig. 1. Motion information is used to advect a set of particles (green circles) in accordance with deformations observed in real-world images from an initial state (a) to a deformed state (c). The particles are positioned on a regular grid in the area surrounding the manipulator. Our approach enables the configuration of a physics simulation so that it replicates the same observed behavior when deforming a virtual object (b,d).

We observe the behavior of an object using a dense optical flow algorithm [4] (see Fig. 1a,c) and match the observed motion to a position-based physics simulation that uses meshless shape matching (MSM) [5] (see Fig. 1b,d). This particular simulation method has been widely used due to its computational efficiency, which is a crucial requirement for a wide range of robotics applications. We show through experiments that the proposed method can correctly estimate the deformability of the foams with different stiffness.

Apart from being complementary to haptic sensors, our purely visual method can also be used to learn meaningful object state changes from demonstration [6], to track unknown objects without a rigidity assumption [7], and to interact with non-graspable objects, such as liquids.

The main contributions are:

- a framework to estimate the deformability of objects based on visual observation and physics simulation,
- a method to match visual observation with a position-based dynamics simulation algorithm.

The paper is structured as follows. In Section II we give an overview of related work. In Section III we describe the method in detail. Next, in Section IV, we present a qualitative evaluation of the proposed framework. Finally, we conclude in Section V.

II. Related Work

There has been tremendous effort within the computer graphics and computer vision communities in simulating the deformation of objects [8], [9]. The most commonly used simulation techniques are mesh-based methods such as Finite Element Method (FEM) [10], [11], [12], mass-spring [13] and boundary element method (BEM) [14], [15]. FEM is able to model the deformation with high physical accuracy but it is computationally expensive and complex, which makes it unsuitable for many time-critical applications [5]. On the other hand, BEM and mass-spring methods are more efficient but limited in the realism of the simulation. The former can be applied only to objects with a homogeneous interior, whereas the latter do not guarantee physically accurate results [9].

In recent years, simulation techniques based on position-based dynamics have received more attention. They update the behavior of objects relying solely on positional information unlike force-based approaches such as FEM and mass-spring methods [16]. They are designed to be fast, controllable and stable in simulating interactive graphical environments such as clothes, deformable solids and fluids [17]. All those properties are desirable assets for active perception in robotics applications. MSM is one of the key works in the area of position-based methods and allows the simulation of deformable objects [5]. It has been used in a wide range of interactive graphical applications [18], particularly for modeling the deformation of human body parts [15]. However in none of these approaches, real world observations are taken into account and the parameters are defined manually. There have been some previous attempts in the spirit of our method. For example, [19] estimates the deformability of mesh-based objects but does so in a purely virtual environment. Alternatively, [20], [21], [22] take real world observations into account to simulate plausible deformations in a virtual environment but the object properties are again defined manually. We instead propose to learn the parameters by simulating the observed behavior.

There exists a wide spectrum of approaches in robotics for learning object properties. The predominant methodology is based on haptic sensors [23], [24], [25], [26]. An interesting work [27], proposes the design of feature descriptors to capture the properties of semi-solid objects and to recognize objects from haptic observation in a supervised manner. In addition, some work has been done on extracting such properties using visual perception alone [28], [29]. There are also methods that combine visual perception with haptic feedback to estimate deformable object properties [30]. All these methods rely on direct interaction with the objects. However, the robot interaction may result in an undesirable permanent deformation of the object. We are convinced that this can be avoided by incorporating a simulation component that can learn the properties of the object by observing a human or another robot and predict its future behavior by using this knowledge. Frank et al. [31] move in this direction by incorporating a FEM method in their framework. The robot is able to learn the elasticity of an object by interacting with it and can plan an action using the deformation of the object as predicted by the simulation. Since FEM simulations are time-consuming, [31] uses Gaussian process regression to approximate the deformation. Instead, we propose a framework that employs position-based methods to arrive at fast and reliable simulation algorithms that are suitable for interactive perception scenarios. Moreover, [32] tracks deformable objects from RGB-D data based on the observed force by integrating a physics simulator and probabilistic generative model. Unlike the proposed method, they do not explicitly estimate object properties such as deformability.

III. Methodology

We use a physics simulation based on position-based dynamics, *i.e.* MSM, to relate the visually observed deformed object configuration to the initial object configuration. To estimate deformation parameter in the simulation, we minimize the difference between the observed 2D shape of the object that deforms and the deformed shape obtained using the simulation approach. We use a dense optical flow algorithm [4] to establish the correspondences between the shape in consecutive image frames and track the motion of the deformed object. This motion information is used as the basis for comparing the simulation results to the observed deformations.

Our current method models deformations in the image plane only. However, the method can be extended to 3D using *e.g.* 3D flow extracted from RGB-D imagery [33].

We next provide a detailed description of the physics simulation itself, how it is coupled with the observations and how we estimate deformability.

A. Deformation Simulation

MSM is a method that is used in position-based physics simulators to simulate elastic and plastic deformation in a visually plausible manner [5]. It represents the shape of simulated objects by a configuration of elementary nodes or particles, as depicted by green circles in the example of Fig. 1a,c. Unlike mesh-based methods, MSM does not require connectivity information between nodes to define an object's configuration. Instead the configuration at time t is fully defined by the particle positions $\mathbf{x}_i^t \in \mathbb{R}^2$ where $i = 1, 2, 3, ...N$, with N the number of particles in the configuration.

In each time step, the positions and velocities of the particles are first updated without considering interactions or any other internal constraints between the particles inside the object. Instead only external forces such as gravity or collisions with the environment are considered. Figure 2 illustrates the effect of this initial update where the initial particle configuration $\mathbf{x}_i^0$, see Fig. 2A, is transformed into an intermediate configuration $\bar{\mathbf{x}}_i$, see Fig. 2B.

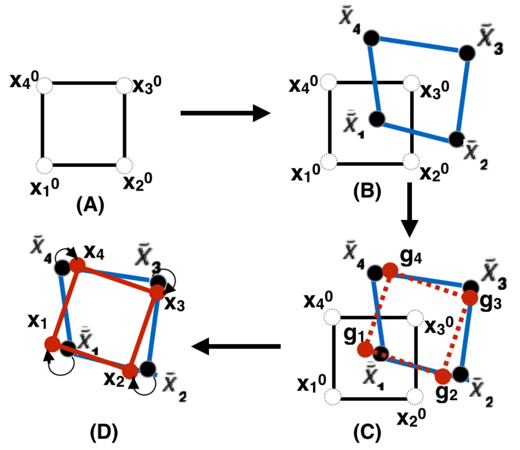


Fig. 2. Meshless shape matching applied to a simple object consisting of four particles. Starting from an initial configuration (A), the particles are moved due to some external force. The updated positions are shown in blue in (B). MSM estimates the optimal transformation that preserves the shape of the initial configuration, as shown in red in (C). In a subsequent integration stage, the particles are moved towards these goal positions (D).

Since these updated particle positions do not incorporate our knowledge of the object shape and its deformability, internal shape constraints are enforced in a subsequent step. Assuming for the sake of exposition that we simply want to maintain the rigidity of the initial shape, as shown in Fig. 2A, we first compute an optimal linear transformation between $\mathbf{x}_i^0$ and $\bar{\mathbf{x}}_i$ and subsequently extract its rotational and translational component. This rotation and translation are then the basis for a rigid transform that is used to move the particles towards a goal position $\mathbf{g}_i$, as illustrated in Figs. 2C,D, that is close to the intermediate configuration, while also respecting the shape constraints. We next discuss these stages in more detail.

The simulation starts by updating the velocity and position of the particles as a result of the force exerted at the current time t

$$\bar{\mathbf{v}}_i = \mathbf{v}_i^{t-1} + \frac{\mathbf{f}_{\text{ext}}^t \Delta t}{m_i} , \tag{1}$$
$$\bar{\mathbf{x}}_i = \mathbf{x}_i^{t-1} + \bar{\mathbf{v}}_i \Delta t , \tag{2}$$

where $\mathbf{f}_{\text{ext}}^t$ is the external gravitational force at t, m_i the particle mass, and Δt the time step between t and $t-1$.

To find an appropriate transformation between the initial configuration $\mathbf{x}_i^0$ and the intermediate deformed configuration $\bar{\mathbf{x}}_i$, the optimal rotation matrix $\mathbf{R} \in \mathbb{R}^{2\times 2}$ is determined by minimizing

$$\sum_i m_i ||\mathbf{R}(\mathbf{x}_i^0 - \mathbf{t}^0) + (\bar{\mathbf{x}}_i - \bar{\mathbf{t}})||^2 , \tag{3}$$

where the optimal translation vectors $\bar{\mathbf{t}} \in \mathbb{R}^2$ and $\mathbf{t}^0 \in \mathbb{R}^2$ are equal to the center of mass of the respective particle configurations

$$\mathbf{t}^0 = \frac{1}{m_c}\sum_i^N m_i \mathbf{x}_i^0 , \tag{4}$$
$$\bar{\mathbf{t}} = \frac{1}{m_c}\sum_i^N m_i \bar{\mathbf{x}}_i , \tag{5}$$
$$m_c = \sum_i^N m_i . \tag{6}$$

To minimize (3), first the optimal linear transformation between the initial and intermediate configuration, $\mathbf{A} \in \mathbb{R}^{2\times 2}$, is calculated as in [5]

$$\mathbf{A} = \left(\sum_i m_i \mathbf{p}_i \mathbf{q}_i^\top\right)\left(\sum_i m_i \mathbf{q}_i \mathbf{q}_i^\top\right)^{-1} = \mathbf{A}_r \mathbf{A}_s , \tag{7}$$

where $\mathbf{p}_i = \bar{\mathbf{x}}_i - \bar{\mathbf{t}}$ and $\mathbf{q}_i = \mathbf{x}_i^0 - \mathbf{t}^0$ are the particle locations relative to the center of mass. The matrix $\mathbf{A}_s$ is symmetric and contains only scaling, no rotation. The rotational part can be obtained by decomposing $\mathbf{A}_r$ into the rotation matrix $\mathbf{R}$ and symmetric matrix $\mathbf{S}$ using polar decomposition as in [5]

$$\mathbf{A}_r = \mathbf{R}\mathbf{S} . \tag{8}$$

The goal positions, see Fig. 2C, can now be determined as

$$\mathbf{g}_i = \mathbf{R}\mathbf{q}_i + \bar{\mathbf{t}} . \tag{9}$$

The shape constraints are incorporated into the simulation by correcting the intermediate particle positions as follows:

$$\mathbf{x}_i^t = \bar{\mathbf{x}}_i + \alpha(\mathbf{g}_i - \bar{\mathbf{x}}_i) , \tag{10}$$
$$\mathbf{v}_i^t = \frac{\mathbf{x}_i^t - \mathbf{x}_i^{t-1}}{\Delta t} , \tag{11}$$

where α controls the speed at which the particles converge to the goal positions. The α value affects the stiffness of the model. If $\alpha = 1$, the goal positions immediately become the corrected positions. If $\alpha < 1$, the configuration gradually converges to its goal shape in each time step. In our experiments, $\alpha = 1$.

To increase the deformability of the models, [5] introduces linear deformation such as shear and stretch. A linear deformation can be obtained by combining $\mathbf{R}$ and $\mathbf{A}$ in the goal position computation:

$$\mathbf{g}_i = ((1-\beta)\mathbf{R} + \beta\mathbf{A})\mathbf{q}_i + \bar{\mathbf{t}} , \tag{12}$$

where β represents the degree of deformation, ranging from 0 to 1. The presence of $\mathbf{R}$ in the sum ensures that there is still a tendency towards the undeformed shape. If β approaches 1, the range of deformation increases, whereas if β is close to 0, the object behaves like a rigid object. Figure 3 demonstrates how different values of β affect the behavior of a virtual object. This β parameter is our parameter of interest for defining an object's deformability. Our goal is to estimate it on the basis of the object's observed motion.

To further extend the range of motion, we use a technique called cluster based shape matching [5] and divide the

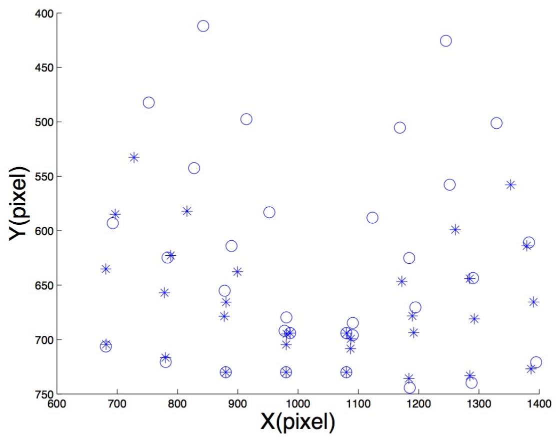


Fig. 3. Effects of the deformation parameter β on the final configuration of a simulated object undergoing a downward external force applied to the center region. Circles and stars correspond to the deformed state with $\beta = 0.1$ and $\beta = 0.9$ respectively. Larger β's result in more deformation.

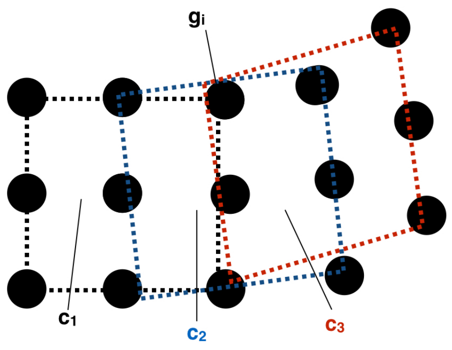


Fig. 4. Example particle cluster configuration with overlapping clusters of size 3×3. Particle i belongs to all three clusters c_1, c_2, and c_3. Consequently its goal position $\mathbf{g}_i$ is set to be the average of the goal positions computed for each cluster separately, namely $\mathbf{g}_i^1$, $\mathbf{g}_i^2$, and $\mathbf{g}_i^3$.

particles into overlapping clusters as shown in Fig. 4. At each integration step, each cluster's initial configuration is matched with its current shape. For each particle which has more than one goal position, the average of the goal positions is computed before the integration to calculate velocity and position [34]

$$\mathbf{g}_i = \frac{1}{N_i} \sum_{j \in \mathbf{c}_i} \mathbf{g}_i^j \ , \tag{13}$$

where N_i is the number of clusters that particle i belongs to, $\mathbf{c}_i$ is the set of clusters particle i belongs to, and $\mathbf{g}_i^j$ is the goal position which is associated with cluster $j \in \mathbf{c}_i$. The cluster size in our implementation is set to 3×2 particles.

In this work we investigate the effects of specific manipulations exerted on a particular part of the object, as illustrated in Fig. 5. To simulate the effects of such manipulations, we select the particle closest to the manipulated part, p_f, and fix its position in accordance with the disturbance, $\mathbf{x}_f$. We do not consider the problem of collision detection between the tip of the manipulator and the object. Instead $\mathbf{x}_f$ is determined from the position of p_f in the final frame, after the particle advection step. See for example the particle marked with a cross in Fig. 5(a). As a result of the manipulation, its position will be fixed to $\mathbf{x}_f$ as shown in Fig. 5(b). We also increase the mass of particle p_f in order to pull the center of mass of the particle cluster towards $\mathbf{x}_f$. Although p_f is included in the shape matching, its goal position is fixed at $\mathbf{x}_f$. Algorithm 1 provides an overview of a single time step of the simulation. For clarity, we omitted the clustering part. Our implementation is based on Müller's publicly available 2D implementation of MSM [35].

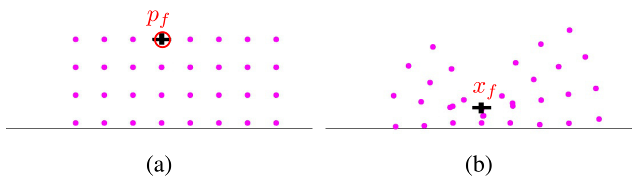


Fig. 5. A specific manipulation applied on a particular particle in the simulation environment. The manipulated particle is labeled as p_f (a). The particle position is constrained by fixing it to a specific position $\mathbf{x}_f$ (b).

B. Coupling Observation and Simulation

To observe the deformation of a real-world object, we estimate its motion in consecutive frames of a manipulation video using a dense optical flow algorithm [4], as implemented in [36].

We use the optical flow to advect a set of particles spread across the object surface. The particles are initiated on a regular 4×8 grid, spaced 100 pixels apart, in accordance with the object boundary as observed in the initial frame, before deformation, as depicted in Fig. 6(a). Note that we only focus on a small area around the contact point of the manipulator and the object. Since the foam objects that we use in the experiments have primitive rectangular shapes, we selected the boundaries of this region in an ad-hoc manner. The object is segmented manually and the particles are placed uniformly in the segmented region. We depict the observed particle positions in this initial configuration by $\tilde{\mathbf{x}}_i^0 \in \mathbb{R}^2$.

To estimate the trajectory of each particle over the image sequence, we use a particle advection approach similar to [37]. A particle's observed position at frame t, $\tilde{\mathbf{x}}_i^t$, is advected to the next frame position $\tilde{\mathbf{x}}_i^{t+1}$ according to the estimated optical flow:

$$\tilde{\mathbf{x}}_i^{t+1} = \tilde{\mathbf{x}}_i^t + \mathbf{o}_i^t \ , \tag{14}$$

with $\mathbf{o}_i^t \in \mathbb{R}^2$ the optical flow averaged in a 10×10 region around the particle. By repeating this advection for each frame, we arrive at the final observed deformed configuration $\tilde{\mathbf{x}}_i^d$, as depicted by the red circles in Fig. 6b,c.

The inputs for the simulation now consist of the initial configuration $\mathbf{x}_i^0 = \tilde{\mathbf{x}}_i^0$ together with the particle p_f to which our manipulation is applied and whose position is fixed to

Algorithm 1: Simulation time step (without clustering)

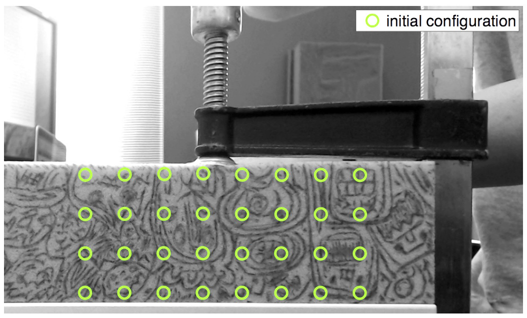
```
Input: $\mathbf{x}_i^0$: initial configuration, $\mathbf{x}_i^{t-1}$: configuration at
       previous time, $\mathbf{v}_i^{t-1}$: velocity at previous time,
       $m_i$: mass, $p_f$: fixed particle index, $\mathbf{x}_f$:
       constraining position, $N$: number of particles,
       $\Delta t$: time step, $\alpha$: stiffness, $\beta$: deformability
Output: $\mathbf{x}_i^t, \mathbf{v}_i^t$
: configuration and velocity at current time
begin
    // external force, Fig. 2B
    for $i = 1$ to $N$ do
        if $(i = p_f)$ then
            // the particle is fixed
            $\bar{\mathbf{x}}_i = \mathbf{x}_f$;
        else
            $\bar{\mathbf{v}}_i = \mathbf{v}_i^{t-1} + \frac{\mathbf{f}_{\text{ext}}^t \Delta t}{m_i}$;
            $\bar{\mathbf{x}}_i = \mathbf{x}_i^{t-1} + \bar{\mathbf{v}}_i \Delta t$;
    // shape matching, Fig. 2C
    compute $\mathbf{t}^0$, the center of mass of $\mathbf{x}_i^0$;
    compute $\bar{\mathbf{t}}$, the center of mass of $\bar{\mathbf{x}}_i$;
    compute the relative positions
    $\mathbf{p}_i = \bar{\mathbf{x}}_i - \bar{\mathbf{t}}$ and $\mathbf{q}_i = \mathbf{x}_i^0 - \mathbf{t}^0$;
    estimate the optimal linear transform
    $\mathbf{A} = (\sum_i^N m_i \mathbf{p}_i \mathbf{q}_i^\top)(\sum_i^N m_i \mathbf{q}_i \mathbf{q}_i^\top)^{-1} = \mathbf{A}_r \mathbf{A}_s$;
    compute the polar decomposition $\mathbf{A_r} = \mathbf{RS}$ with
    $\mathbf{S} = \sqrt{\mathbf{A}_r^\top \mathbf{A}_r}$ and $\mathbf{R} = \mathbf{A}_r \mathbf{S}^{-1}$;
    // goal positions
    for $i = 1$ to $N$ do
        if $i = p_f$ then
            // the particle is fixed
            $\mathbf{g}_i = \mathbf{x}_f$;
        else
            $\mathbf{g}_i = ((1-\beta)\mathbf{R} + \beta \mathbf{A})\mathbf{q}_i + \bar{\mathbf{t}}$;
    // integration, Fig. 2D
    for $i = 1$ to $N$ do
        $\mathbf{x}_i^t = \bar{\mathbf{x}}_i + \alpha(\mathbf{g}_i - \bar{\mathbf{x}}_i)$;
        $\mathbf{v}_i^t = (\mathbf{x}_i^t - \mathbf{x}_i^{t-1})/\Delta t$;
```

$\mathbf{x}_f$. The particle p_f is selected as the one closest to the tip of the tool, see Fig. 6(a). Its position is fixed to the final position observed in the image after deformation.

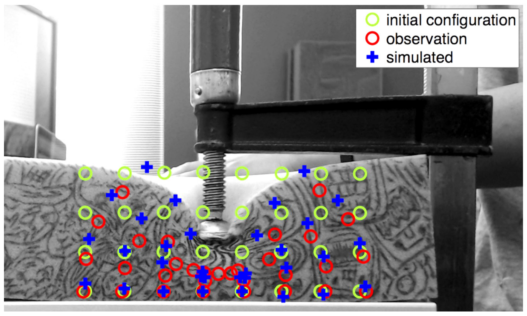
Since we do not include friction force into the model, the simulated object can move in response to the manipulation. To avoid this, we also fix the simulation particles that are on the support surface under the tip and keep the simulated object in the same place as the observed foam. We then run the simulation for a specific β-value until it reaches an equilibrium state with final deformed configuration $\mathbf{x}_i^\beta$.

C. Deformability Estimation

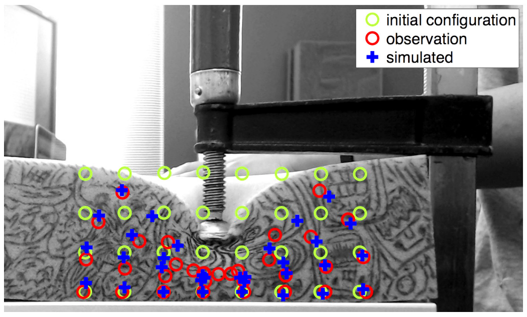
To estimate the deformability parameter β that best describes the deformed object, we minimize an error function that measures the distance between the observed and simu-



(a)



(b)



(c)

Fig. 6. The undeformed object state is shown in (a), with the initial particle configuration depicted by the green circles. The red circles in (b) and (c) correspond to the observed particle positions, advected through the optical flow measurements. The blue crosses are the final deformed configurations obtained from simulating with different β values. In (b), a β value equal to 0.0 was used, whereas in (c), β was set equal to 0.58 as estimated by our approach. The observed and simulated particles are much closer together in (c) than in (b).

lated particles in the final deformed state:

$$E(\beta) = \frac{1}{N} \sum_{i=1}^{N} \| \tilde{\mathbf{x}}_i^d - \mathbf{x}_i^\beta \| \ . \tag{15}$$

To find the minimum, we ran the simulation for a number of β values uniformly sampled from the interval $[0, 1]$. The β with the lowest residual in (15) is selected as representing the objects' deformability.

IV. EXPERIMENTAL EVALUATION

To validate that an approach that combines visual perception with MSM simulation is suitable to characterize the deformability of objects, we investigate the deformations of a

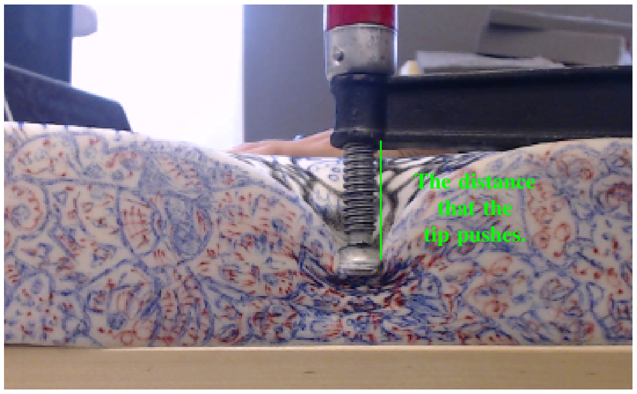


Fig. 7. The flat edge of the clamp's screw is moved to the same end position for each foam.

TABLE I
FOAM PROPERTIES AND ESTIMATED DEFORMABILITY

Foam	1	2	3	4	5
Density	24	28	39	39	50
Hardness	80	150	120	200	230
beta	0.78	0.69	0.60	0.58	0.51

set of foams that are equal in size, but differ in density and hardness. The density is expressed in $\mathrm{kg/m^3}$ whereas the hardness is defined in terms of the force (in N) required to compress the foam to 40%. Together, these parameters determine the firmness. Higher density and hardness imply higher firmness. For instance, *24/80* —with 24 $\mathrm{kg/m^3}$ density and 80 N hardness —is softer than *50/230*. Table I contains the density and hardness of all foams used in the experiments.

To ensure that we squeeze each foam to the same end position, we use a clamp to control the push level, as depicted in Fig. 7. We position a Logitech HD Pro Webcam C920 camera 30 cm away from the foam, to record videos of the manipulations. This particular camera provides a high resolution, a wide field of view, and is adequate to accurately estimate the optical flow of textured objects. The camera's frame rate is 30 Hz and the frame size is 1920×1080 pixels. We apply only normal force using the flat edge of the clamp's screw to capture the 2D motion better. The optical flow is calculated at 10 Hz. The particles are initialized on a 4×8 grid of particles with 100 pixels spacing, see Fig. 6(a), and then advected using the estimated optical flow field. The parameter search is performed as described in section III.C.

A. *Experimental Results*

Table I contains the deformability values β as estimated by our approach for the different foams. We can observe a 0.23 difference between the deformability estimated for the softest (*24/80*) and the firmest (*50/230*) foams. Figure 9 shows these same values as a function of increasing firmness. This figure clearly illustrates that our estimate of deformability is in accordance with the actual characteristics of the foams, and decreases with increasing firmness. This validates that object deformability can be estimated solely using motion

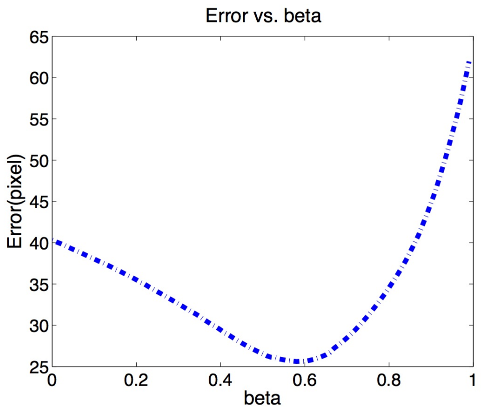


Fig. 8. The error between the observed and simulated particles in the final deformed state of foam *39/200* for different β values. We observe a global minimum around 0.58.

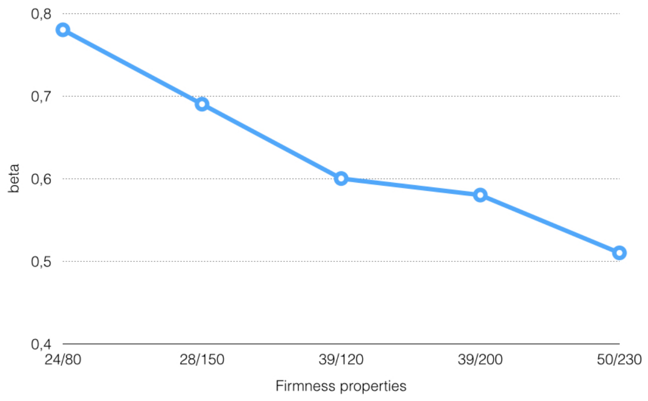


Fig. 9. The optimal β as a function of firmness. The estimated deformability consistently decreases with increasing firmness across all foams.

information and a position-based dynamics simulation using MSM.

We also observe that although the hardness of *28/150* is larger than *39/120*'s, we obtain a larger β for *28/150*, suggesting that our measure of deformability is more sensitive to density than hardness.

We finally show the evolution of the error as a function of β over the $[0, 1]$ interval in Fig. 8. This clearly indicates that there is a single global minimum for the deformability.

V. CONCLUSION

We have presented a framework to estimate the deformability of an object. It relies on optical flow and physics simulation to observe and simulate the behavior of real-world objects and to characterize their deformability. The main novelty of this paper is the use of position-based dynamics using MSM to estimate the deformability of real-world objects based solely on visual perception, thus without knowing the exact force applied. We demonstrated that the framework can correctly distinguish foams with different physical properties.

The estimated deformability can be used directly to adjust the physical properties of a virtual object in the simulation environment so that the real-world object's behavior can be more accurately simulated. This then allows to predict the outcome of an interaction with this object and to avoid undesired behaviors such as permanently deforming or breaking it.

Similar to [38] and [39], we relied on a highly-controlled experimental setup with good visibility and where the object is unoccluded. In our future work, we plan to use a probabilistic approach, such as particle filtering, to consider more realistic scenarios where *e.g.* human hands or robotic grippers occlude the object and the camera viewpoint can be arbitrary.

References

[1] D. Henrich and H. Wörn. *Robot Manipulation of Deformable Objects*. Springer, 2000.

[2] P. Güler, Y. Bekiroglu, X. Gratal, K. Pauwels, and D. Kragic. What's in the container? Classifying object contents from vision and touch. In *IROS*, 2014.

[3] K. Pauwels and D. Kragic. Simtrack: A simulation-based framework for scalable real-time object pose detection and tracking. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Hamburg, Germany, 2015.

[4] J. Pérez, S. Javier, E. Meinhardt-Llopis, and G. Facciolo. Tv-l1 optical flow estimation. *Image Processing On Line*, 2013:137–150, 2013.

[5] M. Müller, B. Heidelberger, M. Teschner, and M. Gross. Meshless deformations based on shape matching. In *SIGGRAPH*. ACM, 2005.

[6] A. Pieropan, G. Salvi, K. Pauwels, and H. Kjellström. Audio-visual classification and detection of human manipulation actions. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Chicago, Illinois, 2014.

[7] A. Pieropan, N. Bergström, M.Ishikawa, and H. Kjellström. Robust 3D tracking of unknown objects. In *ICRA*, 2013.

[8] M. Salzmann and P. Fua. Deformable surface 3d reconstruction from monocular images. *Synthesis Lectures on Computer Vision*, 2(1):1–113, 2010.

[9] A. Nealen, M. Müller, R. Keiser, E. Boxerman, and M. Carlson. Physically based deformable models in computer graphics. In *Computer Graphics Forum*, volume 25, pages 809–836. Wiley Online Library, 2006.

[10] J. Teran, S. Blemker, V. Hing, and R. Fedkiw. Finite volume methods for the simulation of skeletal muscle. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 68–74. Eurographics Association, 2003.

[11] T. McInerney and D. Terzopoulos. A finite element model for 3d shape reconstruction and nonrigid motion tracking. In *Computer Vision, 1993. Proceedings., Fourth International Conference on*, pages 518–523. IEEE, 1993.

[12] S. Ilić, M. Salzmann, and P. Fua. Implicit meshes for effective silhouette handling. *International Journal of Computer Vision*, 72(2):159–178, 2007.

[13] K. Waters and D. Terzopoulos. Modelling and animating faces using scanned data. *The Journal of Visualization and Computer Animation*, 2(4):123–128, 1991.

[14] D.L. James and D.K. Pai. Artdefo: accurate real time deformable objects. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 65–72. ACM Press/Addison-Wesley Publishing Co., 1999.

[15] B. Zhu, L. Gu, J. Zhang, Z. Yan, L. Pan, and Q. Zhao. Simulation of organ deformation using boundary element method and meshless shape matching. In *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pages 3253–3256. IEEE, 2008.

[16] J. Bender, M. Müller, M.A. Otaduy, M. Teschner, and M. Macklin. A survey on position-based simulation methods in computer graphics. In *Computer Graphics Forum*, volume 33, pages 228–251. Wiley Online Library, 2014.

[17] M. Macklin, M. Müller, N. Chentanez, and T.-Y. Kim. Unified particle physics for real-time applications. *ACM Transactions on Graphics (TOG)*, 33(4):104, 2014.

[18] Y. Tian, Y. Yang, X. Guo, and B. Prabhakaran. Haptic-enabled interactive rendering of deformable objects based on shape matching. In *Haptic Audio Visual Environments and Games (HAVE), 2013 IEEE International Symposium on*, pages 75–80. IEEE, 2013.

[19] J. Liu and G. Su. Multi-scale method for adaptive mesh editing based on rigidity estimation. In *Computer Vision, Graphics & Image Processing, 2008. ICVGIP'08. Sixth Indian Conference on*, pages 55–62. IEEE, 2008.

[20] K.A. Sidorov and A.D. Marshall. Learnt real-time meshless simulation. In *Computer graphics forum*, volume 33, pages 147–156. Wiley Online Library, 2014.

[21] S..I Park and J.K. Hodgins. Capturing and animating skin deformation in human motion. *ACM Transactions on Graphics (TOG)*, 25(3):881–889, 2006.

[22] A. Jain, T. Thormählen, H.-P. Seidel, and C. Theobalt. Moviereshape: Tracking and reshaping of humans in videos. *ACM Transactions on Graphics (TOG)*, 29:148, 2010.

[23] S. Chitta, J. Sturm, M. Piccoli, and W. Burgard. Tactile sensing for mobile manipulation. *Robotics, IEEE Transactions on*, 27(3):558–568, 2011.

[24] V. Chu, I. McMahon, L. Riano, C.G. McDonald, Q. He, J.P.-T. Martinez, M. Arrigo, N. Fitter, J.C. Nappo, T. Darrell, et al. Using robotic exploratory procedures to learn the meaning of haptic adjectives. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3048–3055. IEEE, 2013.

[25] H. Yussof, M. Ohka, J. Takata, Y. Nasu, and M. Yamano. Low force control scheme for object hardness distinction in robot manipulation based on tactile sensing. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 3443–3448. IEEE, 2008.

[26] A. Jain, M.D. Killpack, A. Edsinger, and C.C. Kemp. Reaching in clutter with whole-arm tactile sensing. *The International Journal of Robotics Research*, 2013.

[27] M.C. Gemici and A. Saxena. Learning haptic representation for manipulating deformable food objects. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 638–645. IEEE, 2014.

[28] S.B. Matiacevich, D. Mery, and F. Pedreschi. Prediction of mechanical properties of corn and tortilla chips by using computer vision. *Food and Bioprocess Technology*, 5(5):2025–2030, 2012.

[29] A.R. Fugl, A. Jordt, H.G. Petersen, M. Willatzen, and R. Koch. *Simultaneous estimation of material properties and pose for deformable objects from depth and color images*. Springer, 2012.

[30] P. Boonvisut and M.C. Cavusoglu. Estimation of soft tissue mechanical parameters from robotic manipulation data. *Mechatronics, IEEE/ASME Transactions on*, 18(5):1602–1611, 2013.

[31] B. Frank, C. Stachniss, R. Schmedding, M. Teschner, and W. Burgard. Learning object deformation models for robot motion planning. *Robotics and Autonomous Systems*, 62(8):1153–1174, 2014.

[32] John Schulman, Albert Lee, Jason Ho, and Pieter Abbeel. Tracking deformable objects with point clouds. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 1130–1137. IEEE, 2013.

[33] E. Herbst, R. Xiaofeng, and D. Fox. Rgb-d flow: Dense 3-d motion estimation using color and depth. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 2276–2282, 2013.

[34] A.R. Rivers and D.L. James. Fastlsm: Fast lattice shape matching for robust real-time deformation. In *ACM SIGGRAPH 2007 Papers*, SIGGRAPH '07, New York, NY, USA, 2007. ACM.

[35] M. Müller. Demo:meshfree deformations based on shape matching, 2005. "http://matthias-mueller-fischer.ch/demos/matching2dSource.zip".

[36] G. Bradski. *Dr. Dobb's Journal of Software Tools*, 2000.

[37] K.K. Reddyy. *Action Recognition Using Particle Flow Fields*. PhD thesis, Citeseer, 2012.

[38] B. Frank, R. Schmedding, C. Stachniss, M. Teschner, and W. Burgard. Learning the elasticity parameters of deformable objects with a manipulation robot. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 1877–1883, 2010.

[39] A. R. Fugl, A. Jordt, H. G. Petersen, M. Willatzen, and R. Koch. Simultaneous estimation of material properties and pose for deformable objects from depth and color images. In *Pattern Recognition*, pages 165–174. Springer Berlin Heidelberg, 2012.